IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

OLAJIRE OLAJIDE
28/01/2024

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

Summary of methodologies

-Data Collection through API

-Data Collection with Web Scraping

-Data Wrangling

-Exploratory Data Analysis with SQL

-Exploratory Data Analysis with Data Visualization

-Interactive Visual Analytics with Folium

-Machine Learning Prediction

•Summary of all results

-Exploratory Data Analysis result

-Interactive analytics in screenshots

-Predictive Analytics result

Introduction

- Project background and context

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers
  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

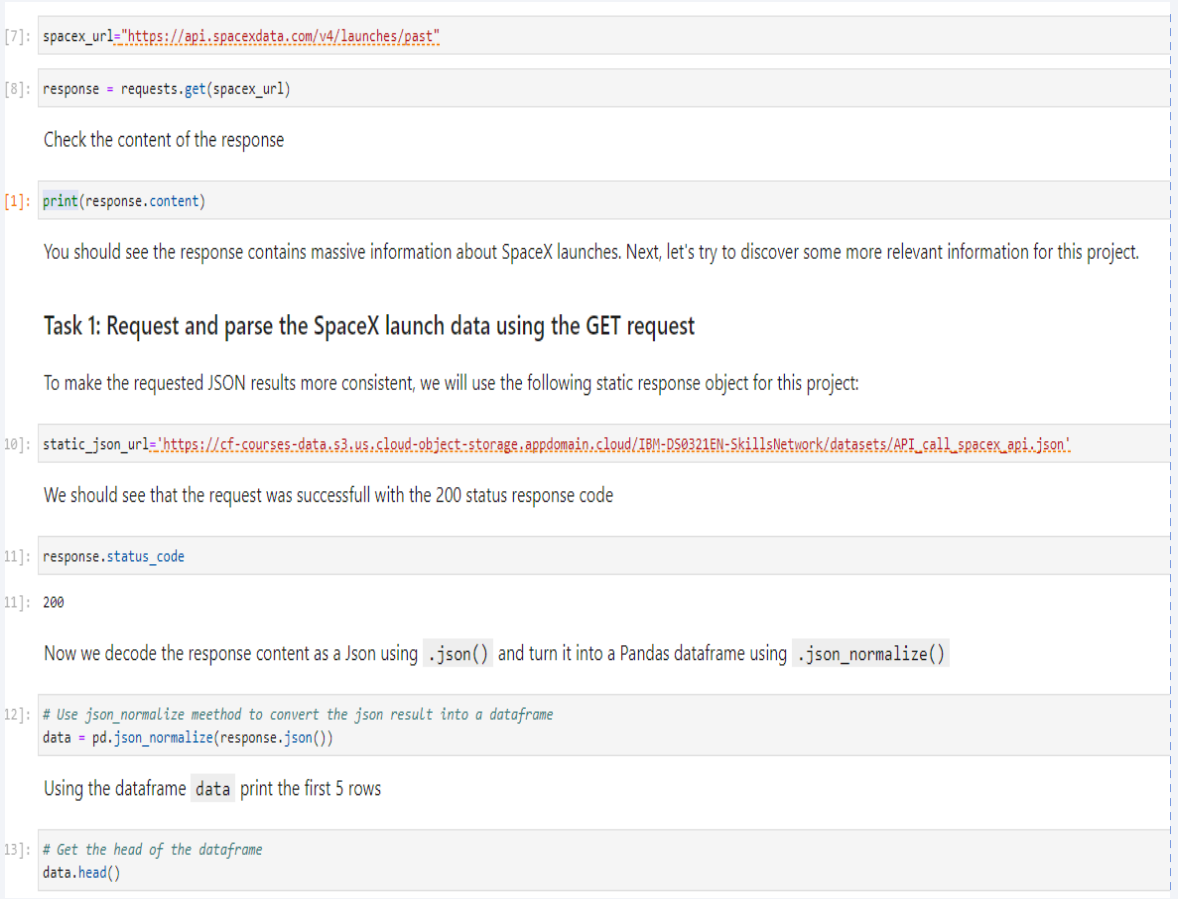
Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API
  - Next, we decode the response content as a json using .json() function call and turn it into a pandas dataframe using .json_normalize()
  - We the clean the data, checked for missing values and fill in missing values where necessary
  - In addition, we performed web scraping from Wikipedia for Falcon 9 lauch records with BeautifuSoup
  - The objective was to extract the launch records as HTML table, parse and convert it to pandas dataframe for future analysis

Data Collection – SpaceX API

- We use the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/jupyter-labs-spacex-data-collection-api%20(1).ipynb



[7]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

[8]:
```
response = requests.get(spacex_url)
```

Check the content of the response

[1]:
```
print(response.content)
```

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

10]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

We should see that the request was successfull with the 200 status response code

11]:
```
response.status_code
```

11]:
```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

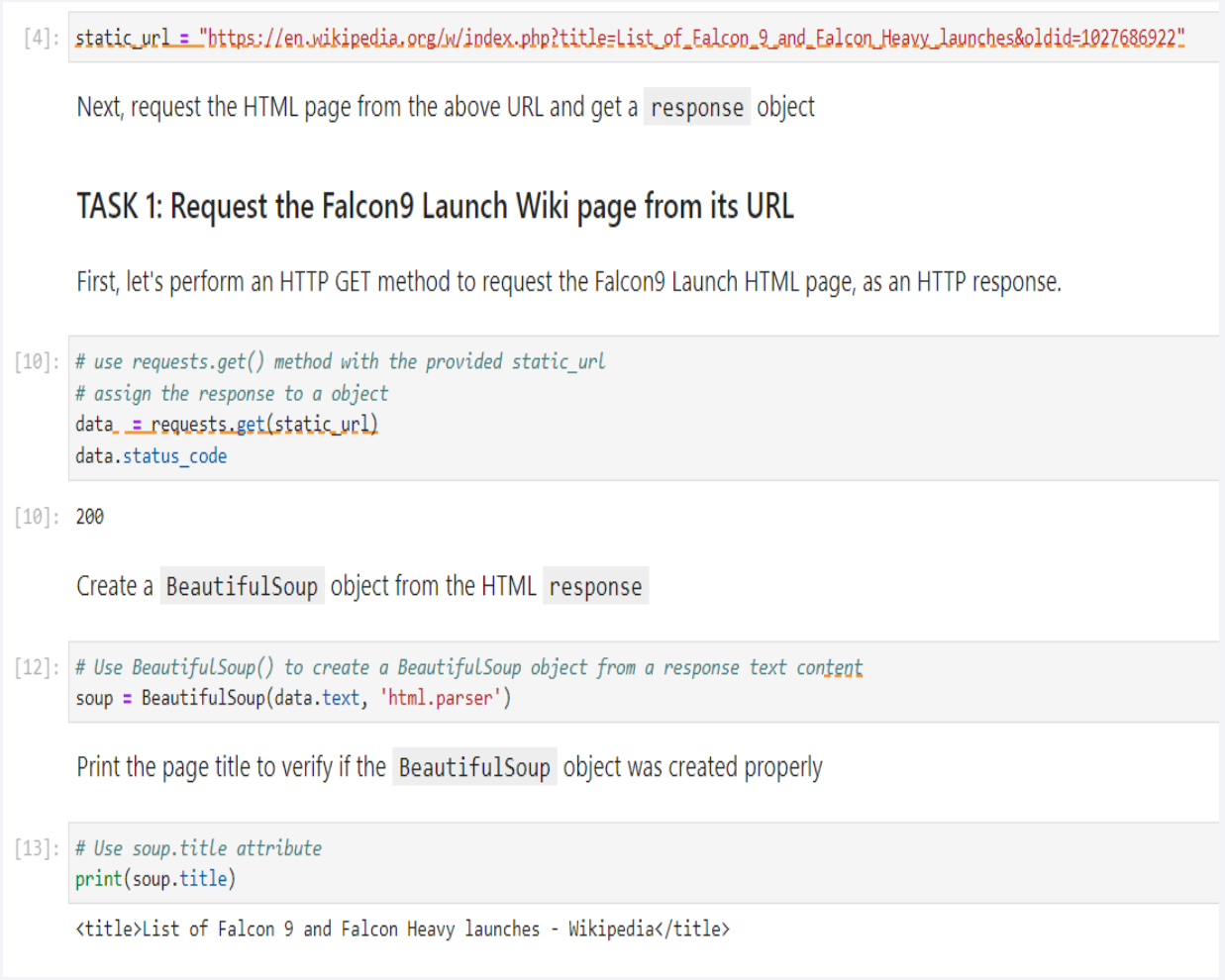
12]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

13]:
```
# Get the head of the dataframe
data.head()
```

Data Collection - Scraping

- We applied webscraping to webscrap Falcon 9 launch records with BeautifulSoup.
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/jupyter-labs-webscraping%20(2).ipynb



[4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

[10]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
data  = requests.get(static_url)
data.status_code
```

[10]:
```
200
```

Create a `BeautifulSoup` object from the HTML `response`

[12]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data.text, 'html.parser')
```

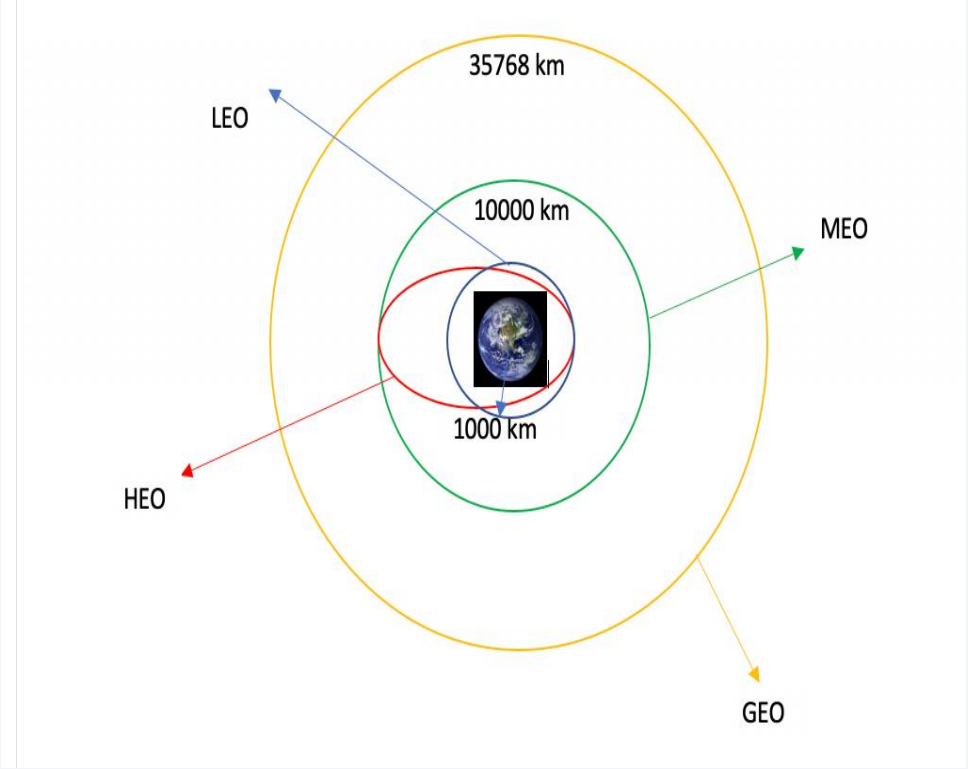
Print the page title to verify if the `BeautifulSoup` object was created properly

[13]:
```
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

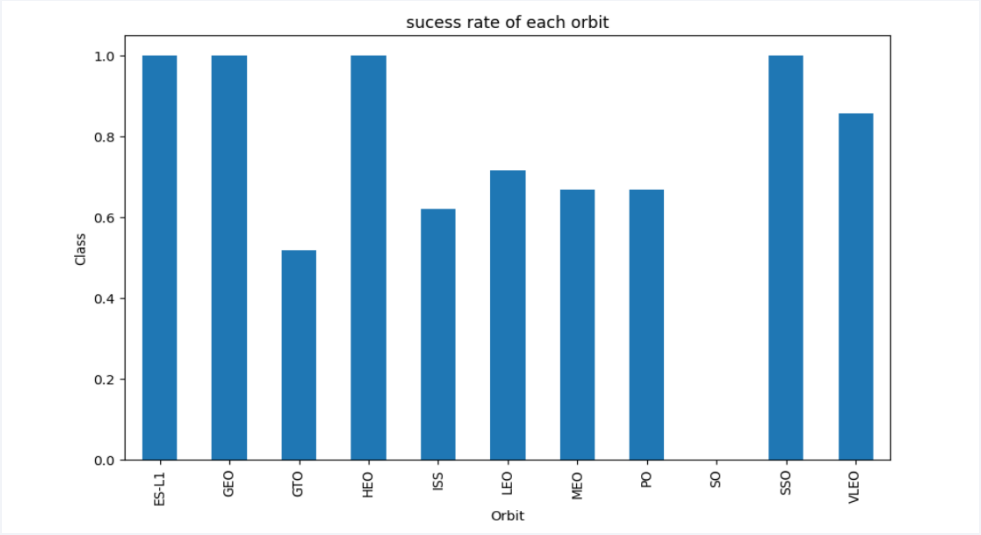
Data Wrangling

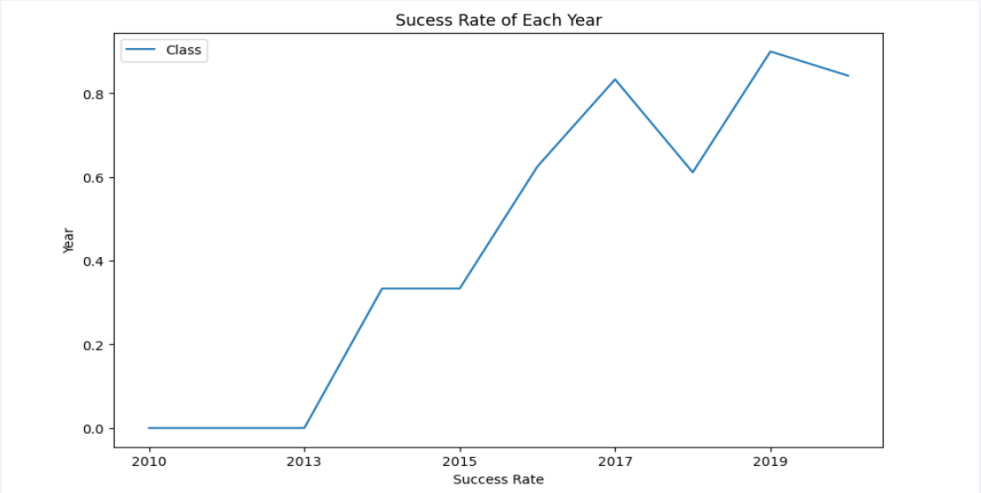
- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits.
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/labs-jupyter-spacex-Data%20wrangling%20(1).ipynb



EDA with Data Visualization

- We explored the data by visualizing the relationship between Payload mass and Flight Number, Launch Site and Flight Number, Launch Site and Payload Mass, Orbit and Flight Number, Orbit and Payload Mass,
- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb





EDA with SQL

- We applied EDA with SQL to get insight from the data. We wrote queries to find out, for instance;
  - The names of unique launch sites in the space mission
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names

- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/jupyter-labs-eda-sql-coursera_sqllite%20(1).ipynb

Build an Interactive Map with Folium

- We marked all Launch sites and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map
- We assigned the feature launch outcomes (failure or success) to class 0 and 1. i.e. 0 for failure and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities.
- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/lab_jupyter_launch_site_location.jupyterlite%20(1).ipynb

Build a Dashboard with Plotly Dash

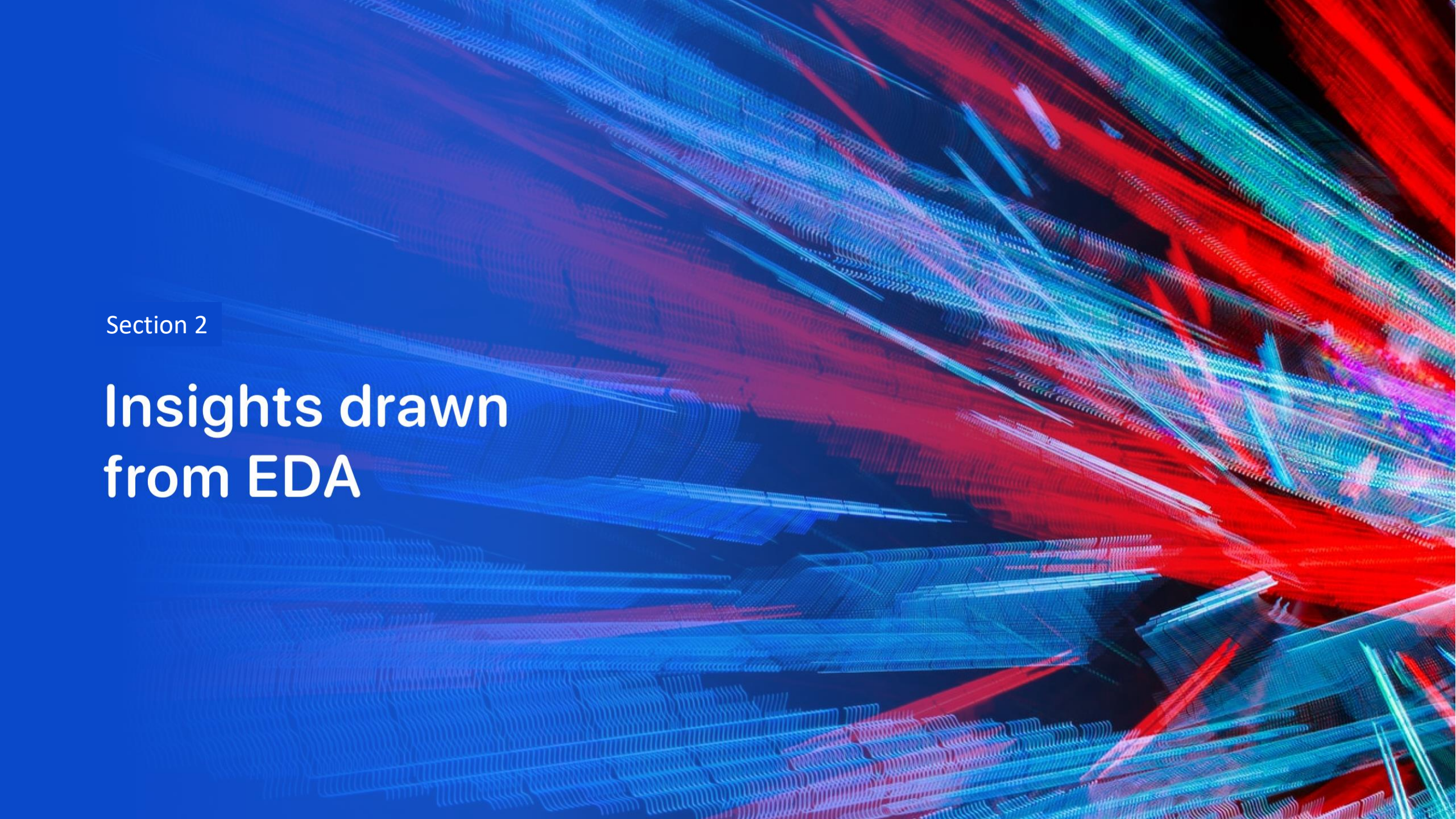
- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (kg) for the different booster version
- Explain why you added those plots and interactions
- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/dashboard.py

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is https://github.com/Ola-Josh/Data_Science_Capstone_SpaceX/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite%20(2).ipynb

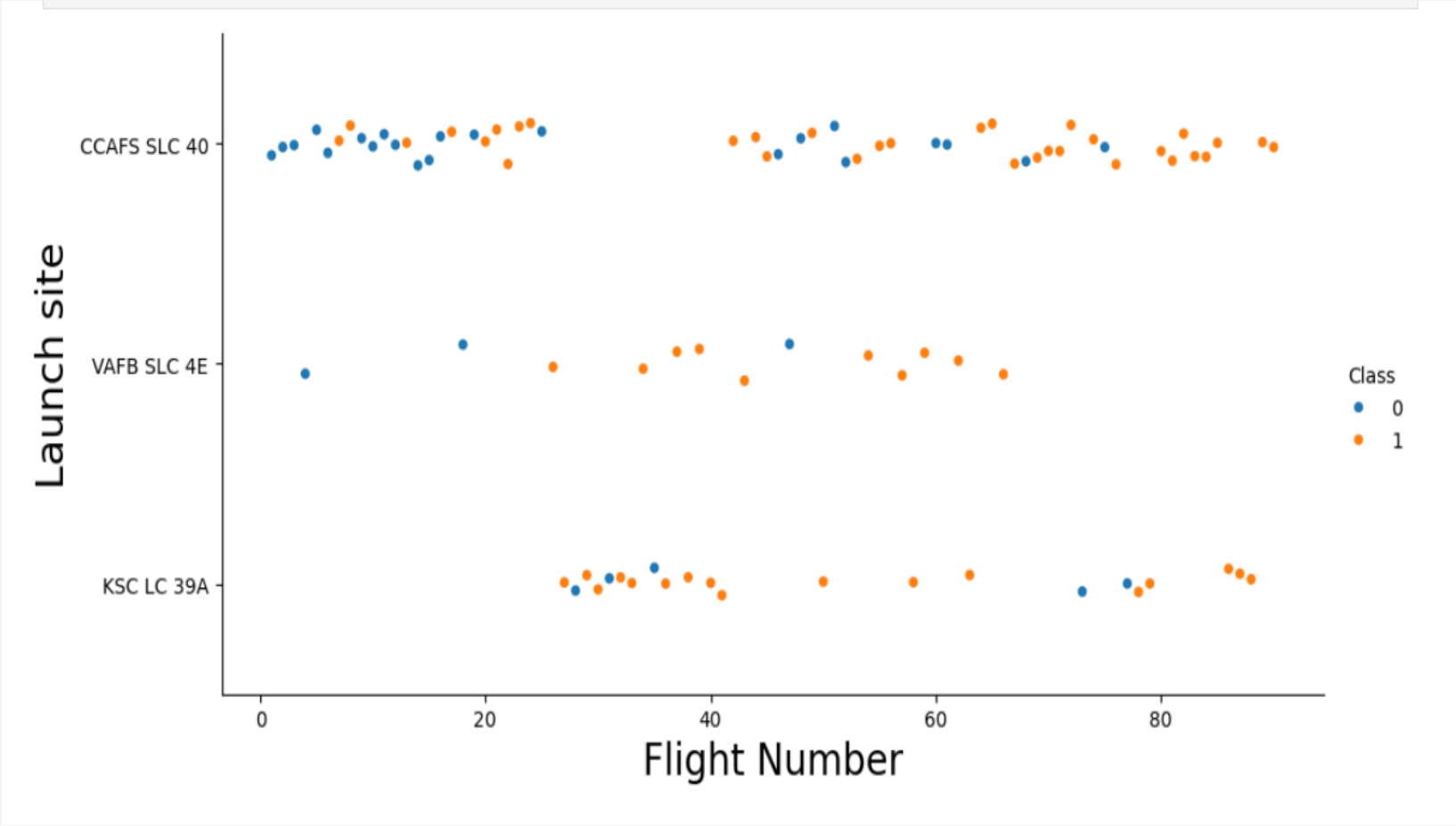
Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results


Section 2
Insights drawn from EDA

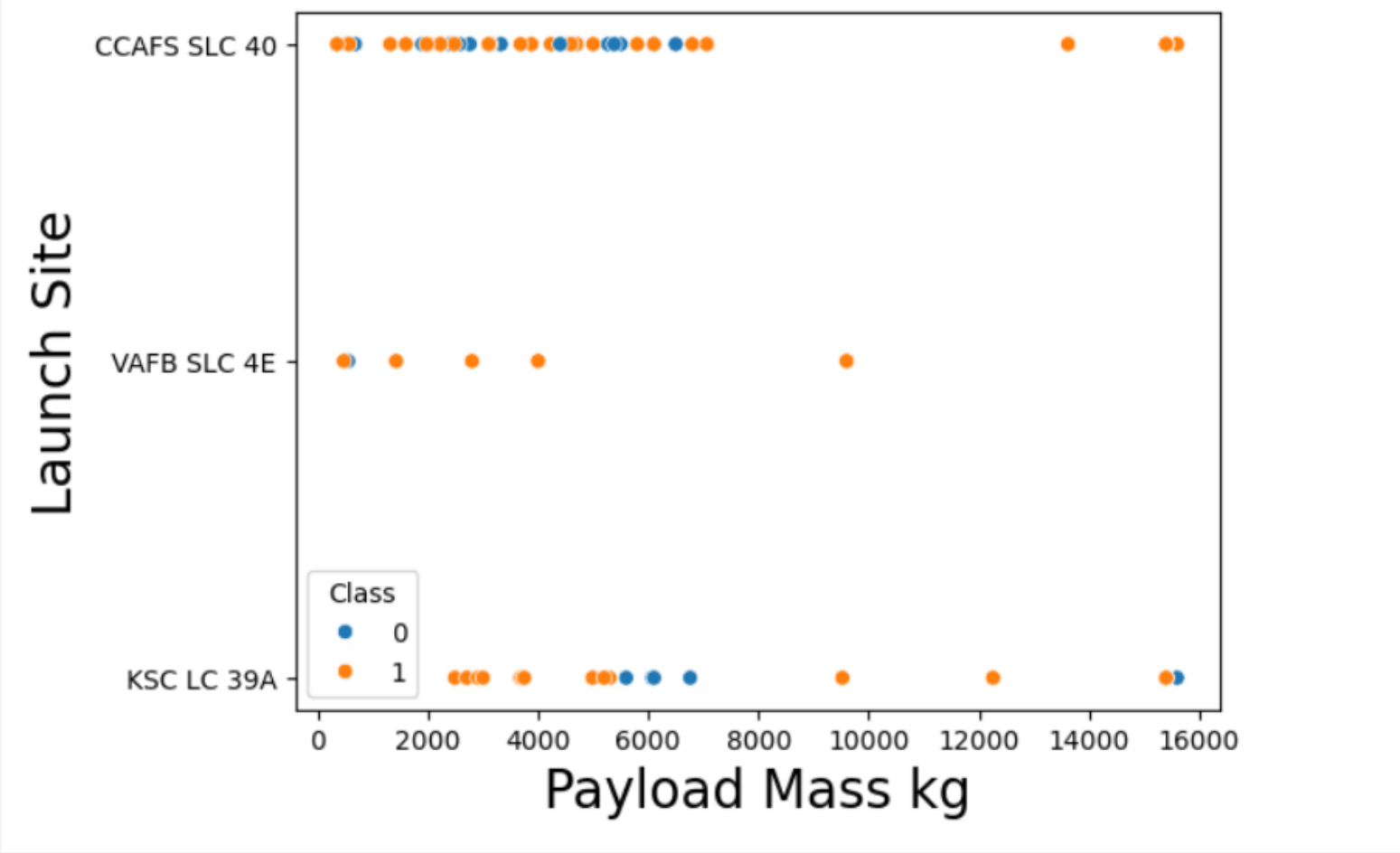
Flight Number vs. Launch Site

- From the plot, we found that the larger the Flight Number at launch site, the greater the success rate at launch site



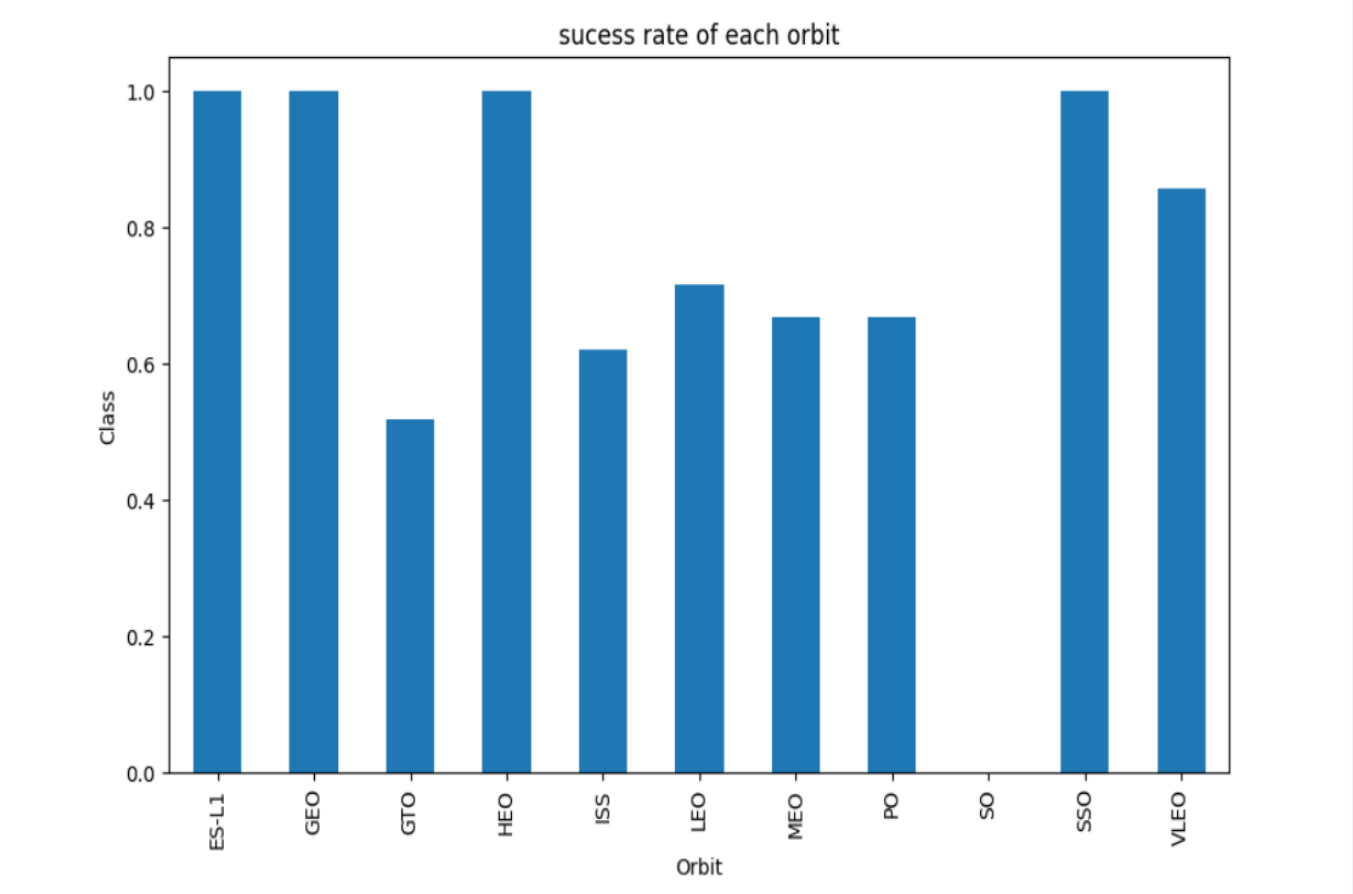
Payload vs. Launch Site

- The greater the Payload Mass for launch site CCAFS SLC 40 the higher the success rate for the rocket.



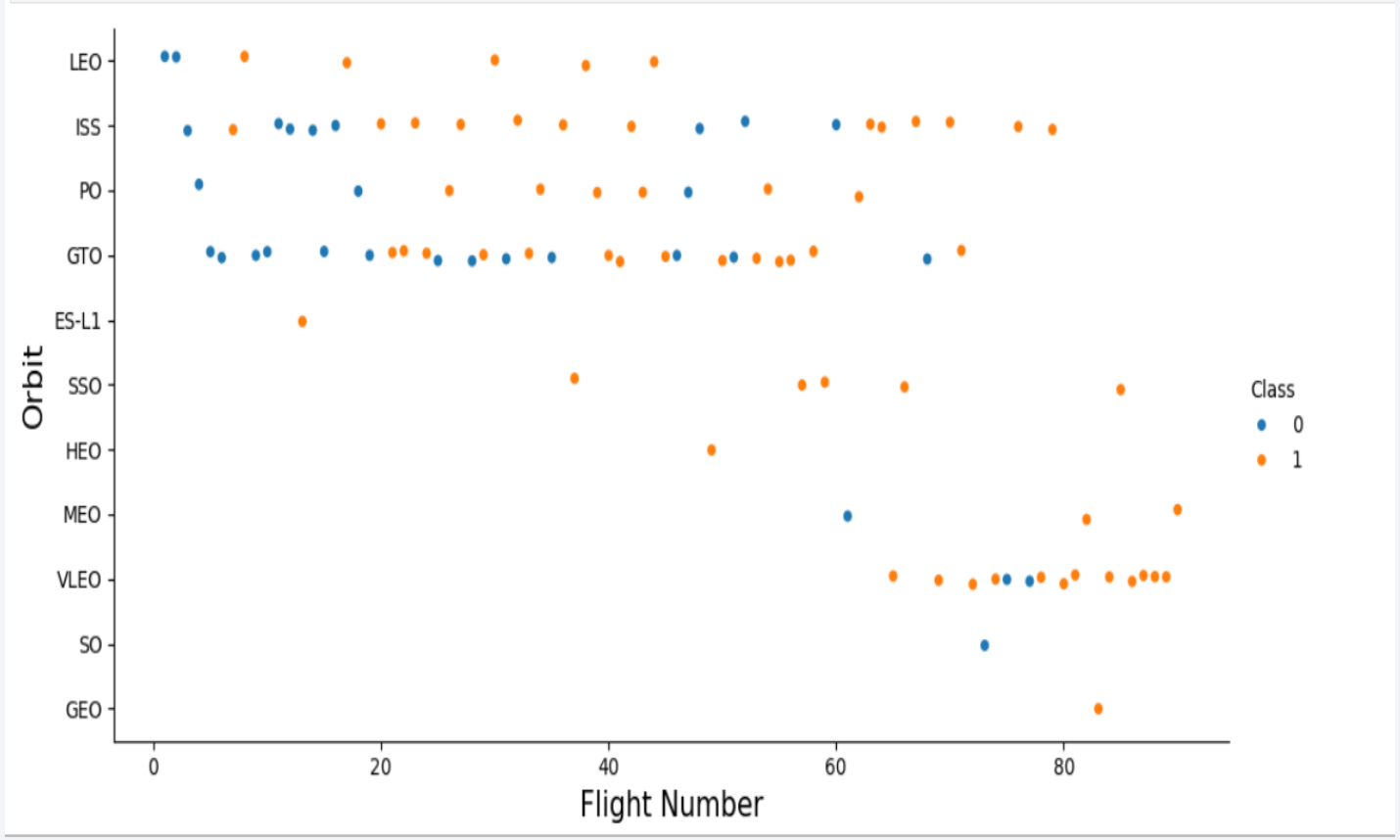
Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type
- Show the screenshot of the scatter plot with explanations



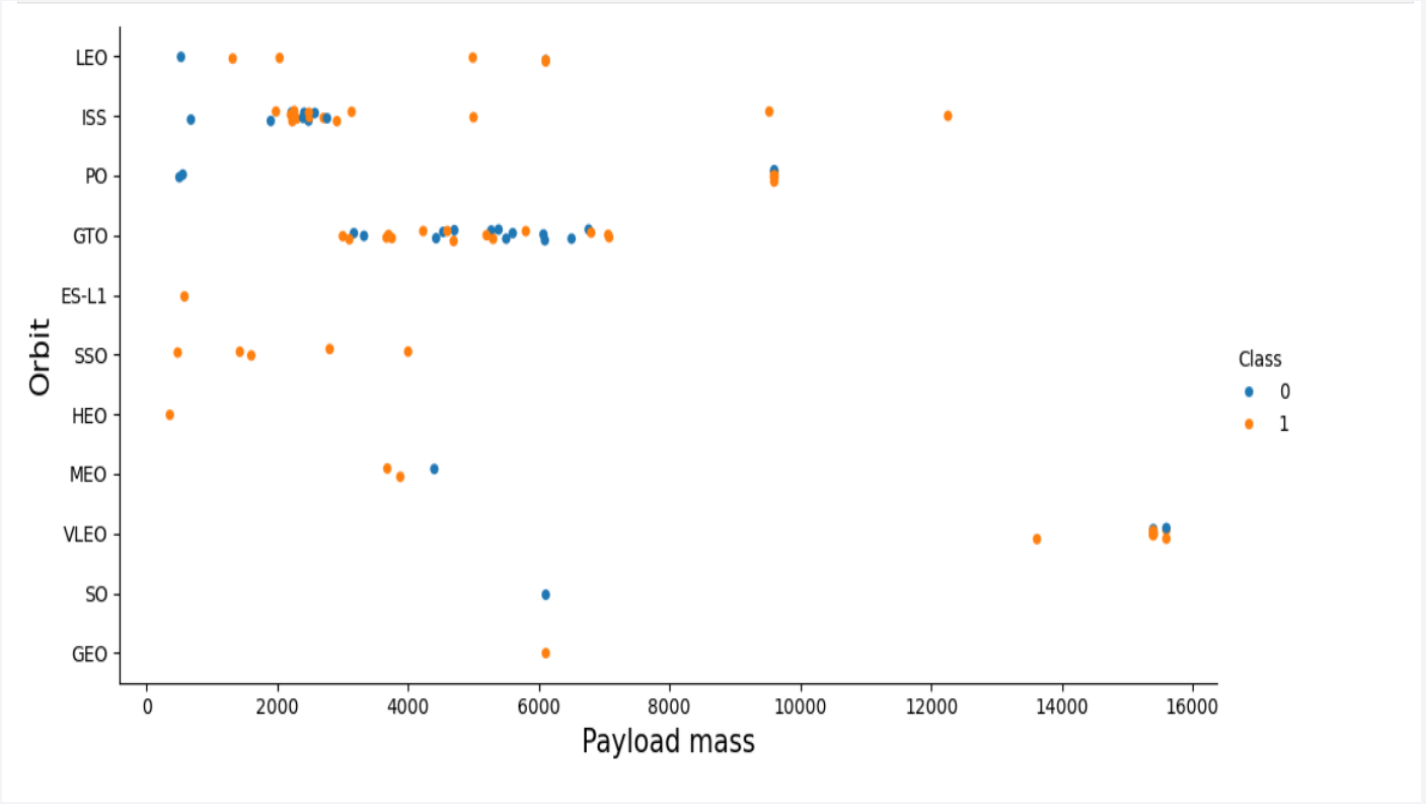
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



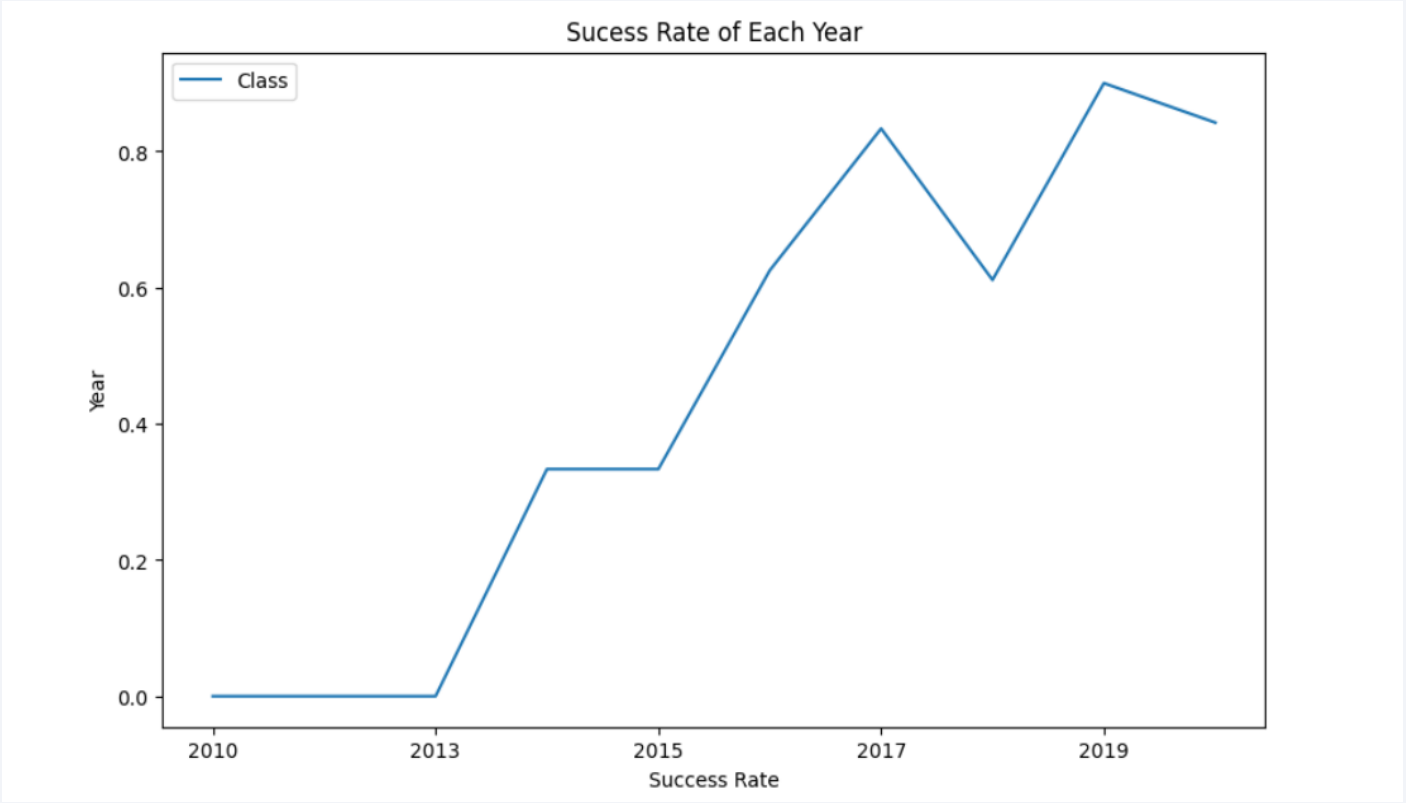
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

- We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

Display the names of the unique launch sites in the space mission

```
40]: #%sql select Unique(LAUNCH_SITE) from SPACEXTBL;
#%sql ibm_db_sa://yyy33800:dwNKg8J3L0IBd6CP@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/BLUDB?security=SS
%sql SELECT DISTINCT (LAUNCH_SITE) FROM SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

40]:

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Launch Site Names Begin with 'CCA'

- We used the query above to display 5 records where launch sites begin with ‘CCA”

```
[9]: %sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
 * sqlite:///my_data1.db
Done.
```

[9]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 45596 using the query below

Display the total payload mass carried by boosters launched by NASA (CRS)

```
[10]: %sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

[10]:

payloadmass
619967

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

Display average payload mass carried by booster version F9 v1.1

```
[20]: %sql select avg(PAYLOAD_MASS__KG_) as avg_payloadmass from SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

[20]:

avg_payloadmass
6138.287128712871

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 4th July 2010

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
12]: %sql select min(DATE) from SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

12]:

min(DATE)
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

[39]:
```
%sql SELECT BOOSTER_VERSION \
FROM SPACEXTBL \
WHERE Landing_Outcome = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 \
AND PAYLOAD_MASS__KG_ < 6000
```

```
 * sqlite:///my_data1.db
Done.
```

[39]:

Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

[14]:
```
%sql SELECT MISSION_OUTCOME, COUNT(*) as total_number \
FROM SPACEXTBL \
GROUP BY MISSION_OUTCOME;
```

```
 * sqlite:///my_data1.db
Done.
```

[14]:

Mission_Outcome	total_number
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

- The total number of Mission outcomes were selected using **GROUP BY**

Boosters Carried Maximum Payload

- We determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

[27]:
```
%sql SELECT BOOSTER_VERSION, PAYLOAD_MASS__KG_ \
FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

```
 * sqlite:///my_data1.db
Done.
```

[27]:

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

- We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

[33]:
```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE, Landing_Outcome \
FROM SPACEXTBL \
WHERE LANDING_OUTCOME = 'Failure (drone ship)' \
AND Date BETWEEN '2015-01-01' AND '2015-12-31'
```

```
 * sqlite:///my_data1.db
Done.
```

[33]:

Booster_Version	Launch_Site	Landing_Outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
[37]: %sql SELECT Landing_Outcome, COUNT(Landing_Outcome) \
FROM SPACEXTBL \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY Landing_Outcome \
ORDER BY COUNT(Landing_Outcome) DESC
```

```
 * sqlite:///my_data1.db
Done.
```

[37]:

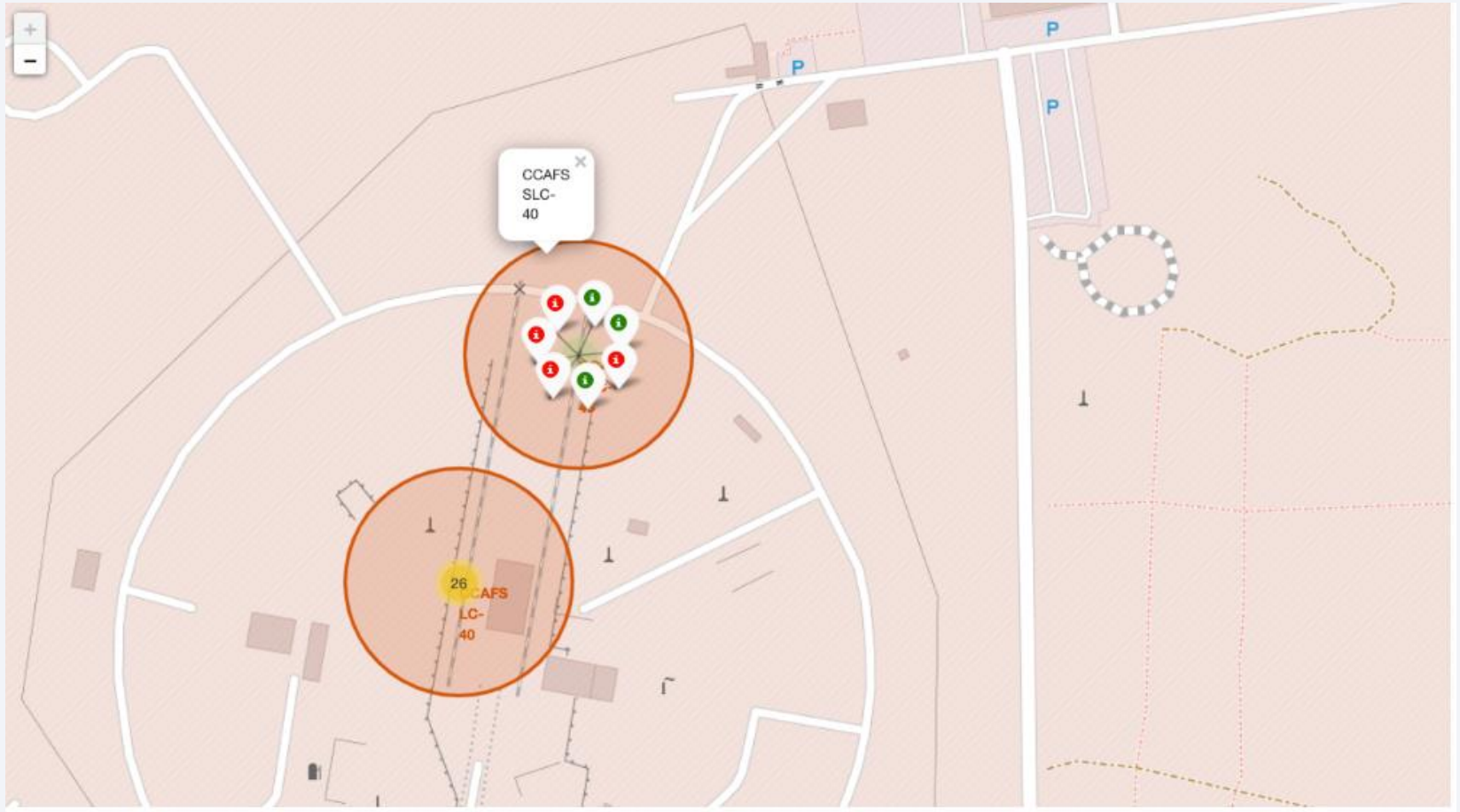
Landing_Outcome	COUNT(Landing_Outcome)
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

All launch sites global map markers

Markers showing launch sites with color labels



Green Marker shows successful Launches and Red Marker shows failures

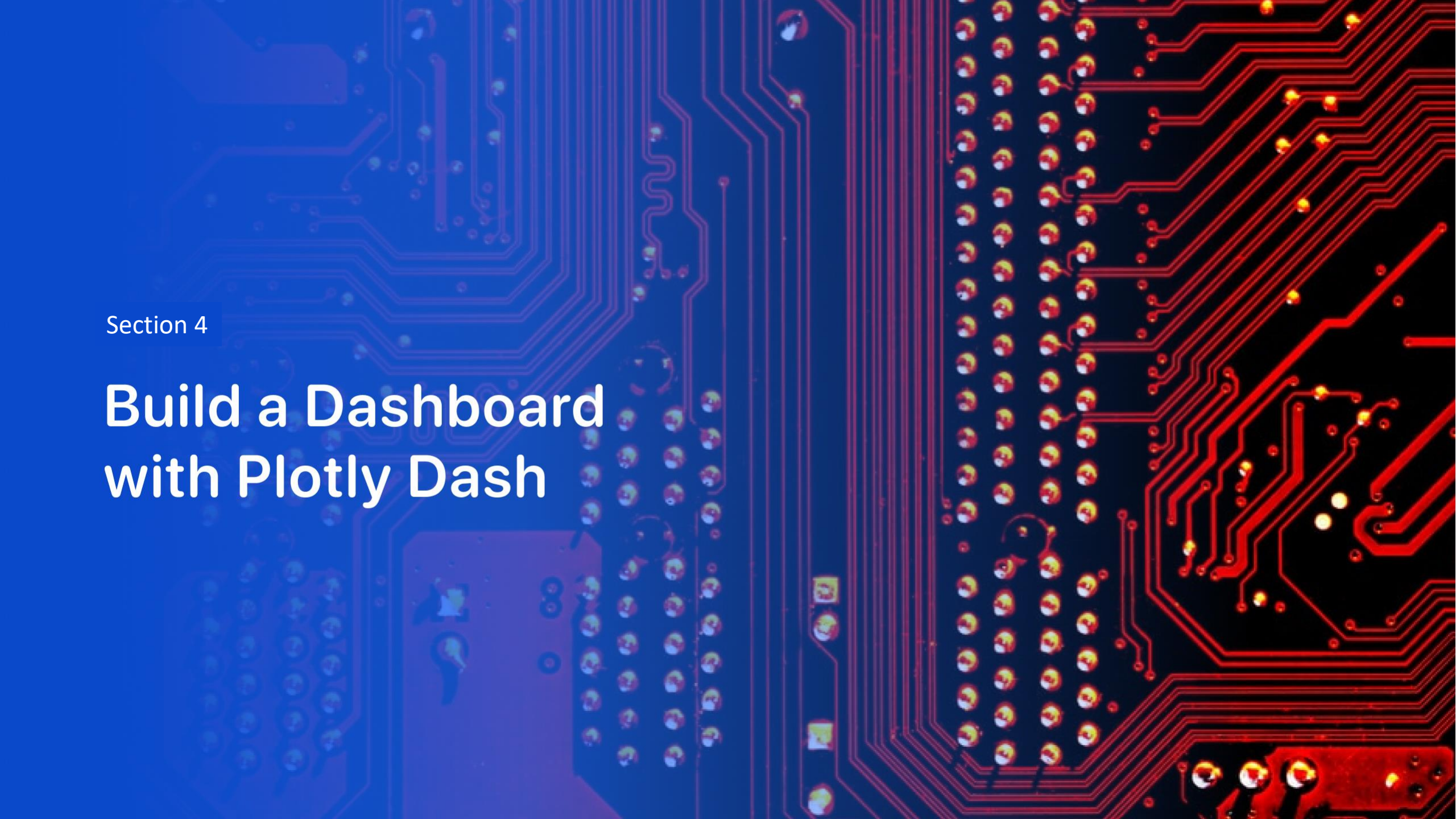
Launch Site distance to landmarks

A railway map symbol may look like this:

A highway map symbol may look like this:

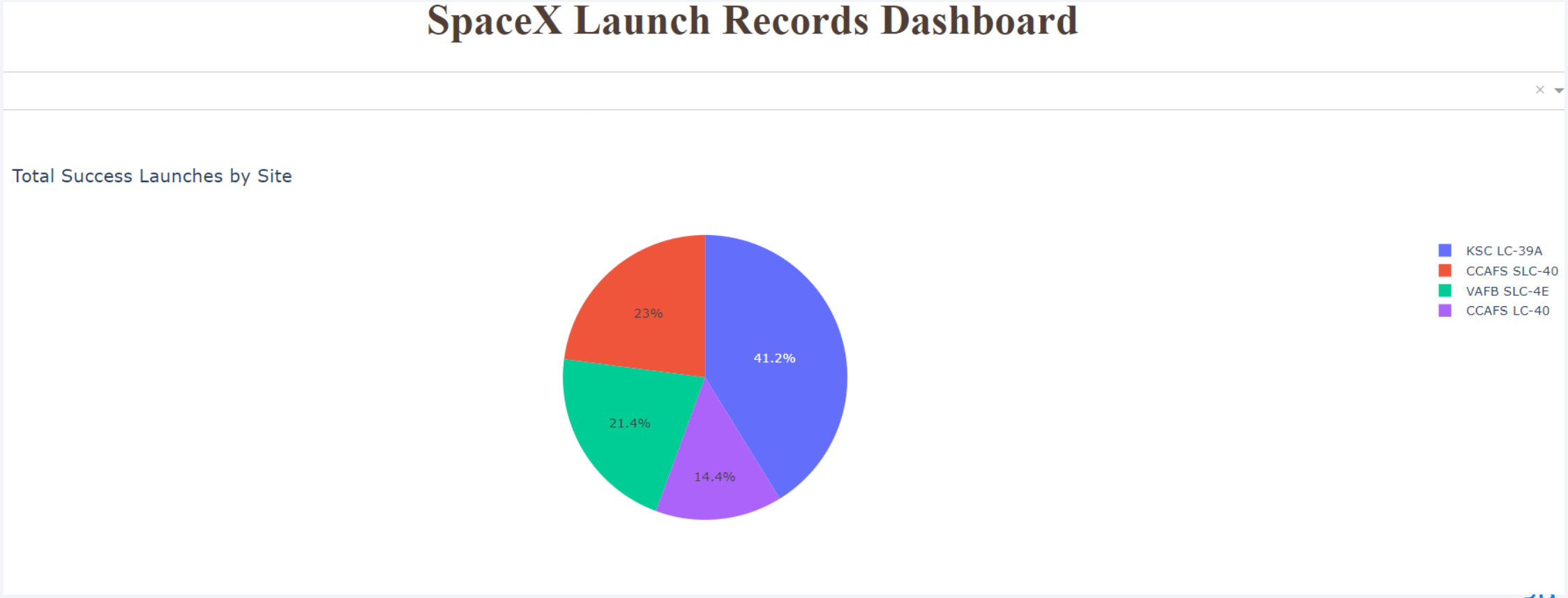
A city map symbol may look like this:

- Are launch sites in close proximity to railways? **No**
- Are launch sites in close proximity to highways? **No**
- Are launch sites in close proximity to coastline? **Yes**
- Do launch sites keep certain distance away from cities? **Yes**

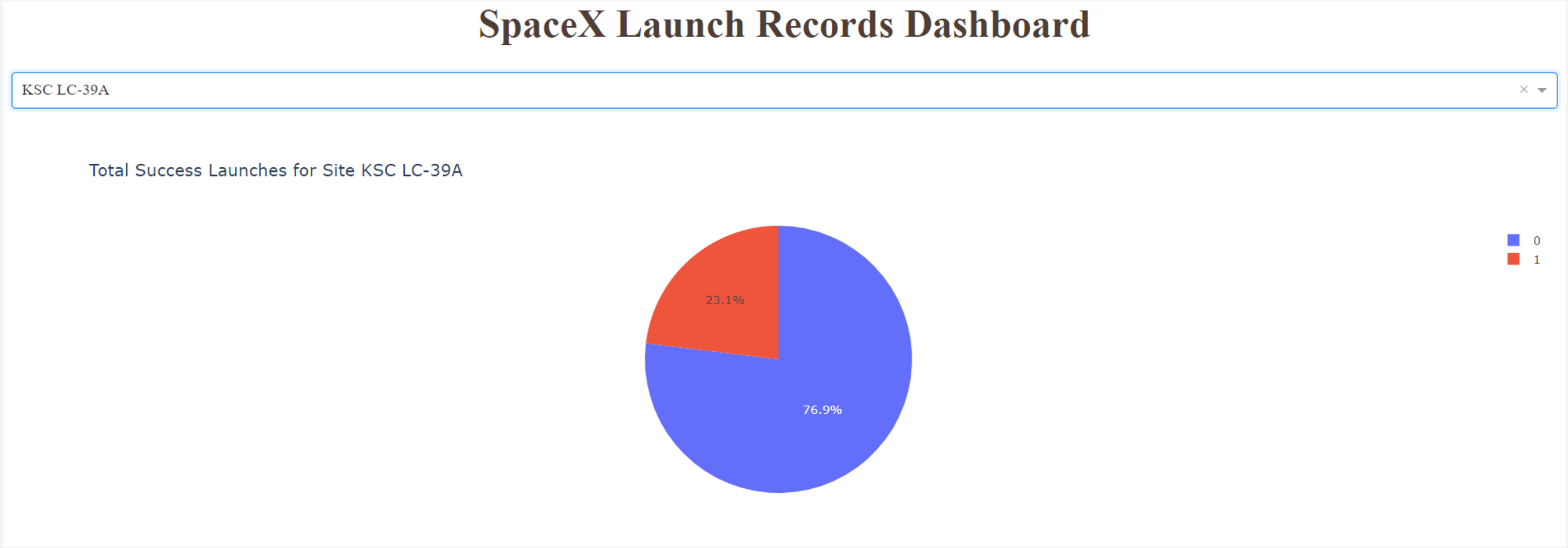

Section 4
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site

We can see that KSC LC-39A had the most successful launches from all the sites



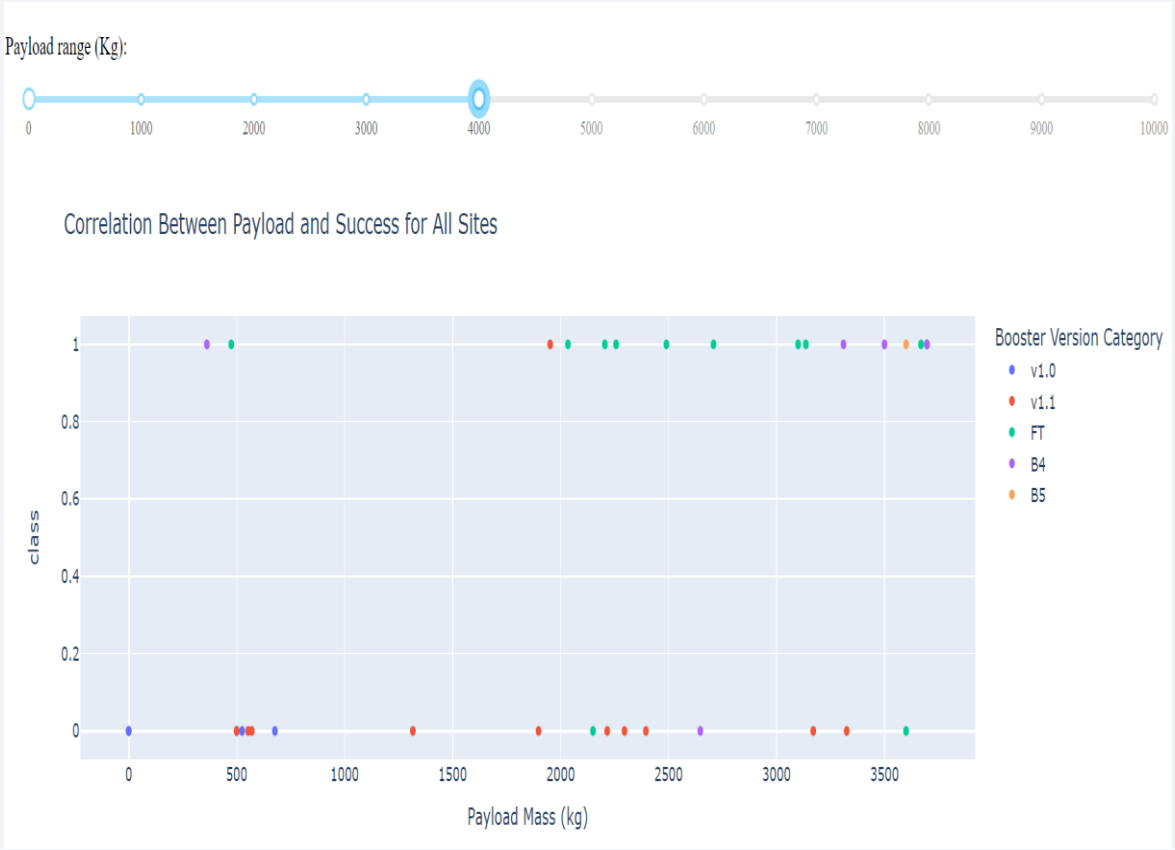
Pie chart showing the Launch site with the highest launch success raio



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg-10000kg

We can see the success for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Find the method performs best:

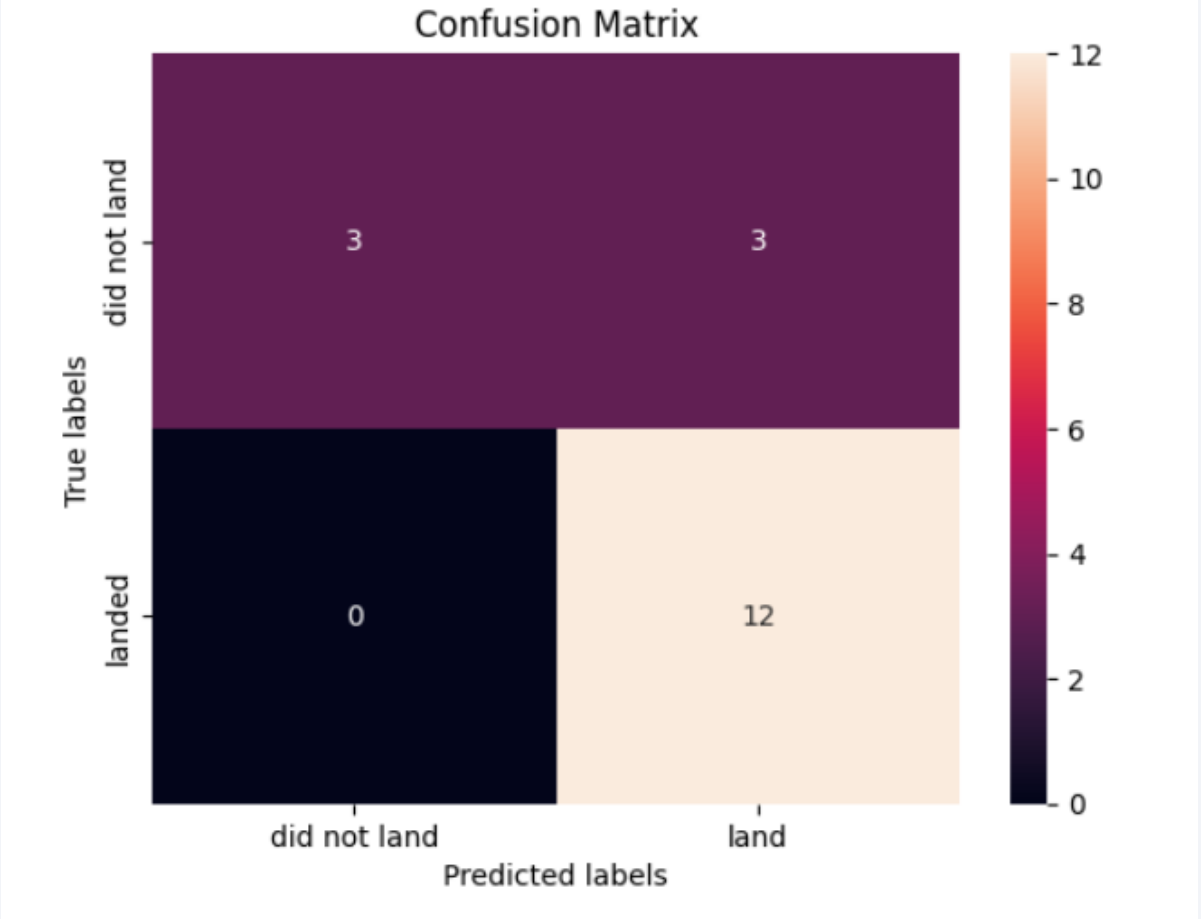
[34]:
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.875
Best Params is : {'criterion': 'gini', 'max_depth': 18, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples
_split': 10, 'splitter': 'random'}
```

- **The decision tree classifier is the model with the highest classification accuracy**

Confusion Matrix

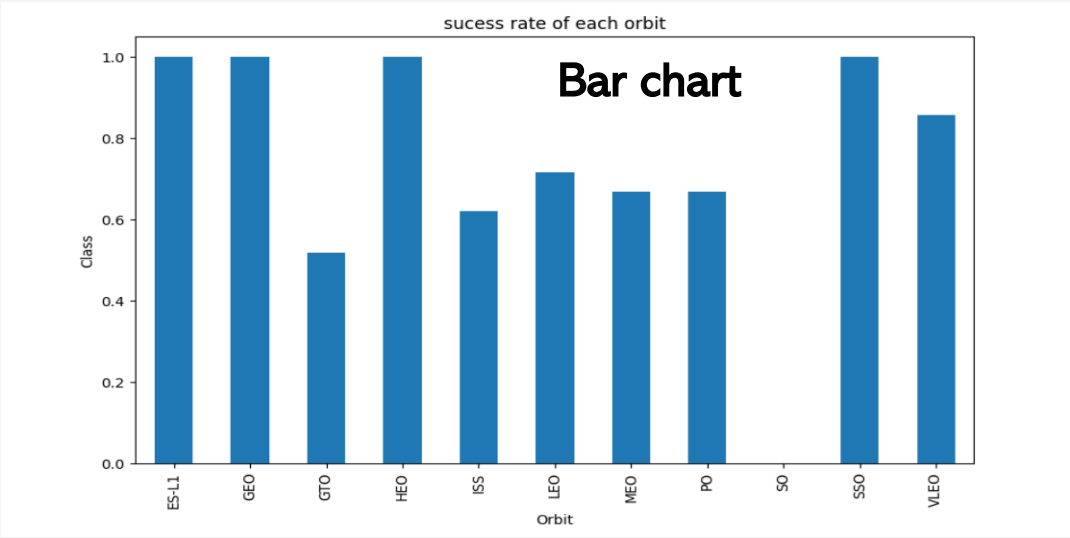
- Examining the confusion matrix, we see that logistic regression can distinguish between the different classes. We see that the major problem is false positives.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Appendix



List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
12]: %sql select min(DATE) from SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

12]:

min(DATE)
2010-06-04

SQL query

Load the dataframe

Load the data

Notebook output

```
[5]: from js import fetch
     import io

     URL1 = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv"
     resp1 = await fetch(URL1)
     text1 = io.BytesIO((await resp1.arrayBuffer()).to_py())
     data = pd.read_csv(text1)
```

```
[6]: data.head()
```

[6]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

dle Mode: Command Ln 4, Col 130 SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Thank you!